Low-Cost Static Gesture Recognition System Using MEMS Accelerometers

Ajay Kannan[α], Ateendra Ramesh[β], Lakshminarasimhan Srinivasan[ψ], Vineeth Vijayaraghavan[δ]
Undergraduate Student[αβψ], Director – Research and Outreach[δ]
College of Engineering, Guindy[α], SSN College of Engineering[β], SRM University[ψ], Solarillion Foundation[δ]
(ajaykannan[α], ateendraramesh[β], lakshminarasimhan.srinivasan[ψ], vineethv[δ])@ieee.org

***Abstract*—The primary objective of the paper is to construct and test a low-cost, minimally supervised gesture recognition system which identifies static gestures efficiently and accurately. The proposed system uses ADXL335 accelerometer sensors which track the gestures and these sensors are interfaced with an Arduino ATMega 2560 micro-controller for data processing and gesture recognition. The software of the system implemented in the micro-controller, features a computationally feasible algorithm which requires only nominal resources to recognize the gestures. The paper further elucidates on minimizing the number of accelerometers to reduce the cost and power-consumption of the system. The performance of the system is assessed using static gestures in the alphabets of the American Sign Language (ASL) across data-sets obtained from 3 trained ASL signers. The average run-time efficiency of the proposed system with a maximum and minimum configuration of 5 and 2 accelerometers was found to be 95.3% and 87.0%, with the cost of these prototype systems being realized at 20 USD and 12.5 USD respectively. It was also found that the system can be trained for the static gestures of the alphabets in ASL under two minutes by a new-user with any system configuration. The authors also feel that the system is compatible with other IoT platforms for interoperability.**

***Keywords*—gesture recognition, accelerometers, sensors, micro-controller, prototype, open source hardware**

I. Introduction

Gesture recognition has been an important field of study among researchers, which revolves around effective and intelligent human-computer interaction. In a broad manner this concept can be explained as the perception of human actions involving the hands, arms, face and the body. The applications based on gesture recognition are vast, ranging from sign language interpretation to modern virtual gaming. However, most available gesture recognition systems are designed for efficient communication and digital interaction of the audio-vocally impaired. This can be attributed to the fact that, around 360 million people worldwide suffer from hearing loss and around 70 million audio-impaired people use sign language as their primal form of communication which was surveyed in the WHO statistics [1] as of March 2015.

For digital interaction, the American Sign Language (ASL) is a well-established and a suitable language, with the alphabets in ASL being described with 24 static gestures and 2 dynamic gestures (*alphabets- 'j' and 'z'*). Factors of sign languages like dynamic motion, two-handed gestures, and speed and orientation of the gesture play a major role in the design and analysis of gesture recognition systems. The design qualities of an ideal gesture recognition system can be elicited as affordable, efficient recognition, robust hardware design, low computation, quick response, generality with respect to recognition of any discrete gesture and also minimal training by an end-user. Commercially available systems that implement these characteristics are expensive. However, the WHO statistics [1] also show a major portion of people with disabling hearing loss live in low and middle income countries to whom such systems are not a viable or an affordable solution. Also recognition models which are inclined towards a vision-based approach proves to be quite complex because of the use of cameras, thus resulting in large data and complicated algorithms such as the use of image processing techniques for feature extraction and processing as well as latency issues [2].

The development of gesture recognition systems was established with the Sayre Glove which was created in 1977. Since then, commercially available gloves such as Data Entry Glove, DataGlove™ and Cyberglove [3] have been used for gesture recognition and they consist of sensors such as flex and contact sensors, gyroscopes and optic-fibre (flexion) sensors. Low-cost approaches utilize sensors developed in a lab environment [4], [5], which require intensive sensor calibration thus making it infeasible on a commercial scale and manually cumbersome for an end-user.

The paper proposes a gesture recognition glove using only accelerometer sensors which would be interfaced with an ATMega 2560 micro-controller, implementing a minimally-supervised low-computational model for accurate and efficient translation of the static gestures. The system is validated and tested using the alphabets of ASL because the American Sign Language being vast in its outreach across various communities is also widely experimented along the field of gesture recognition.

II. Related Work

The domain of gesture recognition devices has been well established over the last few decades, where the glove based systems have been in spotlight among a growing number of researchers. Rung-Huei Liang et al [6] have implemented Hidden Markov Models on a DataGlove™ and Yasir Niyaz Khan et al [7] have developed a sensor based glove implementing an ANN for identifying gestures. These computational models are found to be complex. However, the model prototyped in this paper uses a computationally lightweight algorithm for gesture recognition. Jerome M.Allen et al utilised a Cyberglove [8] to recognize static gestures, while the smart glove developed by

978-1-5090-5249-0/17/$31.00 ©2017 IEEE

Tushar et al [9] uses 14 sensors comprising of hall and bend sensors for gesture recognition. Both these gloves use a large number of sensors thus increasing the system cost and creating multiple points of hardware failure. The system proposed in this paper is low cost, providing the flexibility of choosing between 2 to 5 sensors. The wireless gesture recognition system proposed by Othman Sidek et al [10] requires an external database for mapping and storage. The prototype implemented in this paper requires no external storage for operation. Rishikanth et al [4] and Harini Sekar et al [5] have created gesture recognition systems which use custom made Flex and Contact sensors, which require intensive calibration. The robustness and commercial scalability of these systems can be challenging because of the use of such sensors. However, this paper proposes an alternative by using commercially available accelerometers that does not require any calibration, whilst not compromising on system performance or cost.

III. Problem Statement

Gesture recognition systems for easy and efficient communication of the audio-vocally impaired have been an active research topic. Most commercially available gloves [3] though being efficient, utilize numerous sensors along with complex recognition models, thus making them sophisticated and unaffordable to the bottom of the socio-economic pyramid. Low-cost gesture recognition systems previously proposed use custom-made sensors which require intensive training and sensor calibration, therefore might not being commercially scalable. Hence, there is a need to develop a low-cost yet robust system with off-the-shelf components which also implements a minimally supervised & computationally inexpensive model for efficiently recognizing gestures and interpreting it as alphabets of the English language that can be commercially scalable.

The gesture recognition system prototyped in this paper makes use of commercially available MEMS ADXL335 accelerometer sensors which require no sensor calibration & consumes less power. The data generated by the sensors is passed through a lightweight recognition software embedded in an ATMega 2560 micro-controller for near real-time and accurate translation of static gestures from the alphabets of ASL. The model is also used to test the recognition accuracy while minimizing the number of sensors and simultaneously finding the optimal training required for the system.

IV. Methodology

A. System Hardware Design

Amongst the commercially available MEMS Accelerometers, the ADXL335 was selected as it is low cost, easy to mount and requires only 3-analog connections with less power consumption, taking up to 5V (regulating it to 3.3V using an output pin). The 3 *Analog Outputs* are ratio-metric between 0V to 3.3V corresponding to the acceleration experienced by the sensor with full scaling in between.

For the system operation and control, the ATMega 2560 was chosen because it satisfies the system design requirements of utmost 5 or at least 2 accelerometers corresponding to 15 or 6 analog I/O lines along with adequate computational and memory capabilities.

TABLE I
Interfacing between the sensors and the analog lines from the Micro-controller

Finger	Acc. No.	Output Lines	Analog Lines
Little	A^1	$A^1_{x,y,z}$	Analog 0-2
Ring	A^2	$A^2_{x,y,z}$	Analog 3-5
Middle	A^3	$A^3_{x,y,z}$	Analog 6-8
Index	A^4	$A^4_{x,y,z}$	Analog 9-11
Thumb	A^5	$A^5_{x,y,z}$	Analog 12-14

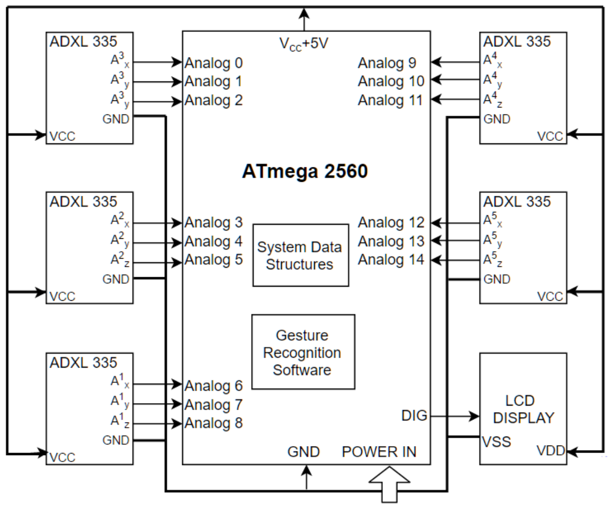

Fig. 1. Functional block diagram for Gesture Recognition Glove at maximum configuration involving 5 accelerometers and the corresponding analog connections as tabulated in Table I.

Gesture Recognition Glove Design: The gesture recognition glove is designed using n_a accelerometers $(2 \leq n_a \leq 5)$ which are interfaced with the micro-controller as shown in figure 1. The accelerometer being a very sensitive sensor results in a differing output even with the slightest change in orientation or placement. Hence, the accelerometers are fixed at the tips of each finger with a rigid support over cotton gloves based on the point of maximum movement in a finger and as shown in figure 2.

B. System Software Design

The $(3 * n_a)$ integer analog-output values generated by the n_a accelerometers as explained in Section IV-A, are stored in an integer temporary array called the Raw-Data Array ($RDArray$). These values are reinitialized every iteration by accepting new values from the analog-pins, which is approximately every 1 millisecond.

1) Initial-Set Array: The average of the $RDArrays$ produced by the accelerometers, corresponding to the open fingered position as shown in figure 2 which is recorded over

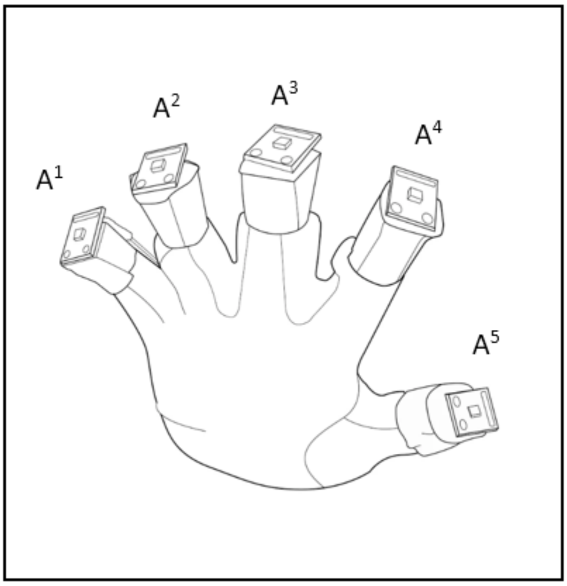

Fig. 2. A right hand in an open fingered position, wearing a Gesture Recognition Glove with 5 accelerometers.

2 seconds and is stored in a new floating point array called the Initial-Set Array ($InitialSetArray$), whose length is also $(3 * n_a)$. This data array serves as a reference for both calibrating the device and processing the gesture data.

$$InitialSetArray = \frac{\sum_{n_s} RDArray}{n_s}$$

Where n_s is the number of samples acquired over 2 seconds ;

2) K-Array and its threshold: A new floating point array called the K-Array ($KArray$) stores the absolute difference between the respective values present in the $RDArray$ and the $InitialSetArray$, which is analogous to the relative position of all the sensors with the initial position.

$$KArray = |RDArray - InitialSetArray|$$

The $KArray$ is calculated every time an $RDArray$ is acquired. An integer constant *delta* (δ), which was empirically found to be 20, is defined as a threshold for the values in the $KArray$, eliminating any minute vibrations or disturbances faced by the accelerometer. When any value in the $KArray$ crosses δ, the start of the gesture is defined and a timer is triggered. This timer is configurable with a preset value of 500ms and is used to signify the end of the gesture.

3) Feature Set Array: A new floating point array called the Feature-Set Array ($FSArray$) holds the final set of features or data points, which define a signed gesture ($FSArray$ is the last recorded $KArray$). The $FSArray$ serves as the input for both the recognition algorithm and a reference look-up table.

4) Reference Set: The Reference Set ($RSArray$) is a floating point array with length $(3 * n_a)$ which consists of the average of the $FSArray$'s for every alphabet signed n_T number of times ($1 \leq n_T \leq 10$), where n_T stands for the number of trials performed by the user. This data structure is calculated during the training phase as explained in the Section IV-C.

$$RSArray = \frac{\sum_{i=1}^{n_T} FSArray[i]}{n_T}$$

5) Look-Up table: The Look-Up table (LUT) is a two-dimensional $\{G \times (3 * n_a)\}$ floating point array which stores $RSArray$'s for all the 24 static gestures of ASL, where G corresponds to the number of gestures, as shown in Table II. It gives no preference to the order in which the alphabets are stored.

$$LookUpTable = \{RSArray(Gesture_1), \ldots, RSArray(Gesture_G)\}$$

6) Gesture Recognition Software: The gesture recognition algorithm employs a simple lightweight approach to finding the closest reference in the LUT with the $FSArray$ acquired when the system operates in Default Mode, which is further elaborated in the Section IV-C. This approach requires low computational overhead and utilization of minimal resources which are within the system's requirements.

Algorithm 1 Manhattan Distance

```
procedure L1(X_1, X_2)
    Sum ← 0
    Len ← length(X_1)
    while i < Len do
        Sum ← Sum + |X_1[i] − X_2[i]|
        i ← i + 1
    end while
    return Sum
end procedure
```

Manhattan Distance - $L1$ Algorithm: This algorithm is used as a metric to determine the closeness of 2 arrays by finding its sum of absolute differences. It is chosen over other higher order distance norms due to its higher efficiency (which is further elaborated in Section IV-D), while only having a time complexity of $O(n_a * G)$.

Algorithm 2 Recognition Module

```
procedure CORE(LUT, FSArray)
    min_1, min_2 ← ∞
    matched_index ← −1
    for R ∈ LUT do
        min_1 = L1(R, FSArray)
        if min_2 > min_1 then
            min_2 = min_1
            matched_index = LUTIndex(R)
        end if
    end for
    print alphabet(matched_index)
end procedure
```

TABLE II
AN INSTANCE OF A LOOK-UP TABLE CREATED BY USER-1 AT REAL-TIME $\{n_a$=5, n_T=2$\}$

Alphabets	A_x^1	A_y^1	A_z^1	A_x^2	A_y^2	A_z^2	A_x^3	A_y^3	A_z^3	A_x^4	A_y^4	A_z^4	A_x^5	A_y^5	A_z^5
A	22	59.5	118	5.5	61.5	122	10	31	134.5	29	25	128	16	15.5	2.5
B	15.5	4.5	4	7.5	4.5	2	6.5	5.5	0.5	4	6	1	18.5	47.5	57
C	3	32	62.5	6	29.5	95	7.5	52.5	89	13	57	83	11	23	15
D	4	20.5	88	4.5	40	112	14.5	9.5	123	9.5	14	5	6	24.5	4
E	4.5	35	88	13	20	125	28.5	6	130	35.5	19	124	14.5	63.5	70
F	11.5	3	2	4	3	0.5	11	15.5	1	22.5	66	78	6.5	53	6.5
G	35.5	3	66.5	52	9.5	70	63.5	10	68.5	77	11.5	59.5	67	43	73.5
H	33	31.5	50.5	50	34	51.5	58	7	72	61	24	65	18.5	43	89
I	16.5	22.5	3	9.5	16.5	133.5	15	8.5	141	43	19	131	30.5	21.5	10.5
K	5	9.5	118	23	6	122	34	45.5	46	30.5	7.5	6.5	17.5	1.5	1.5
L	0.5	11.5	138	15	15.5	138.5	12	32.5	136	20	18	5	32	23	6.5
M	16.5	50.5	88.5	23.5	45	119	13.5	46	125	6	35.5	124.5	6	58	53.5
N	5.5	8	127.5	10	10	129.5	3	38.5	122	3.5	38	118	29.5	43.5	10.5
O	7.5	60.5	83	14	43.5	124	19.5	56	122.5	10	60	114	9	27	0
P	5	70.5	58.5	29.5	56.5	82.5	16.5	9	137	34.5	78	73.5	29	40.5	87
Q	26.5	81.5	38.5	8	69	36.5	20.5	68.5	57.5	17.5	25	134.5	12	16	126.5
R	4.5	19.5	124.5	19.5	20.5	127.5	27.5	40	7	51.5	18.5	12	5.5	48.5	11.5
S	17	11.5	119	20.5	9.5	129.5	21.5	19.5	132	29	15.5	130.5	9	54	24
T	13.5	11.5	125	24	6	126.5	23	13	130.5	23.5	25.5	120	21.5	26.5	2
U	7	43.5	116.5	24	31.5	122	6.5	3.5	3.5	13.5	5	5	25.5	54	7.5
V	17.5	33.5	116	40.5	34	119.5	31	12.5	2	13.5	3	3	11.5	49	25
W	15.5	41.5	92	0.5	17.5	3.5	5.5	8.5	2	8.5	18.5	1	20.5	58.5	16.5
X	10	7	129	29	6.5	130	36.5	7.5	130.5	12	62	70.5	10	64	18.5
Y	37.5	29.5	34	42.5	6	130	46.5	20	129.5	40.5	51.5	115.5	3.5	6.5	1.5

Recogniton Module: The *core procedure* in this module drives the recognition software by calling the above algorithm and determines the closest alphabet in the LUT for display.

C. System Operation

The system operates in two modes, namely the Training mode and the Default mode. The system is set to the Training mode for a new-user, where the system is configured specifically for that user. For all further usage by that user, the system is set to the Default mode. The user also is given the option to vary n_a based on their cost/efficiency requirements which are showcased in the following section.

TABLE III
RECOMMENDED NUMBER OF TRIALS FOR EACH MULTITUDE OF ACCELEROMETER, USING L1 ALGORITHM.

n_A	Recommended n_T
5	2
4	3
3	5
2	7

1) Training Mode: Every time a new user uses the system they must configure the system to their personalized settings through a sequence of steps which are minimally supervised. As explained in Section IV-B, once the glove is worn by a new-user, the $InitialSetArray$ is calculated after which the user is given the option to set the value of n_T and configure the timer. From observations as further explained in Section IV-D, the values of n_T are recommended based on the n_a used in glove as shown in Table III. Then the user is prompted to create the LUT after which the system is set to Default Mode.

2) Default Mode: Once the system is set to Default Mode, the system waits for the user to sign. When the user signs the gesture, the currently signed $FSArray$ is captured and passed onto the gesture recognition software. The recognized alphabet is displayed in an LCD display.

The complete operation of the system is elaborated with a flowchart as shown in figure 3.

D. Observations and Results

The G static gestures ($G = 24$) comprised of the alphabets in ASL were taken as the *Gesture-Set* for testing the efficiency of the gesture recognition system. The alphabets 'J' and 'Z' are not included in the *Gesture-Set* as the gestures are dynamic in nature, being defined by the entire motion and not just a final posture alone. Each gesture in the *Gesture-Set* was sampled 100 times for a total of $G * 100$ gestures by three trained right hand dominant users for various combinations of n_a and n_T, thus forming $3 * (n_a) * (n_t)$ user-specfic *Data-Sets*. It was observed that each $FSArray$ formed was user specific; thereby each user had to train their respective systems separately.

TABLE IV
EFFICIENCY OF SYSTEM USING L1 ALGORITHM FOR VARIOUS NUMBER OF TRIALS AT EACH CONFIGURATION OF n_a (X). THE HIGHLIGHTED VALUES SIGNIFY THE EFFICIENCIES CORRESPONDING TO THE n_T (Y) VALUES IN TABLE III.

x \ y	1	2	3	4	5	6	7	8	9	10
5	91.4	95.3	96.4	96.9	97.1	97.3	97.5	97.5	97.6	97.7
4	91.7	94.9	95.4	95.8	96.0	96.1	96.2	96.3	96.3	96.3
3	90.5	92.8	93.1	93.4	93.5	93.5	93.6	93.7	93.7	93.7
2	85.3	86.1	86.4	86.7	86.8	86.9	87.0	87.2	87.2	87.2

TABLE V

CONFUSION MATRIX FOR SHOWCASING THE ACCURACY OF THE GESTURE RECOGNITION SOFTWARE; YELLOW: SIGNED ALPHABET, GRAY: RECOGNIZED ALPHABET; GREEN: CORRECT CLASSIFICATIONS, RED: FALSE POSITIVES.

	A	B	C	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
A	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	0	95	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0
D	0	1	0	98	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	98	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
F	0	1	0	0	0	99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	3	97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
K	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0
M	0	0	0	0	1	0	0	0	0	0	0	95	0	4	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	1	99	0	0	0	0	0	0	0	0	0	0	0
O	1	0	0	0	0	0	0	0	0	0	0	0	0	99	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	98	0	0	1	0	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	84	0	0	0	0	0	0
T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0
U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	97	1	0	0	0
V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	4	95	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0
X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100

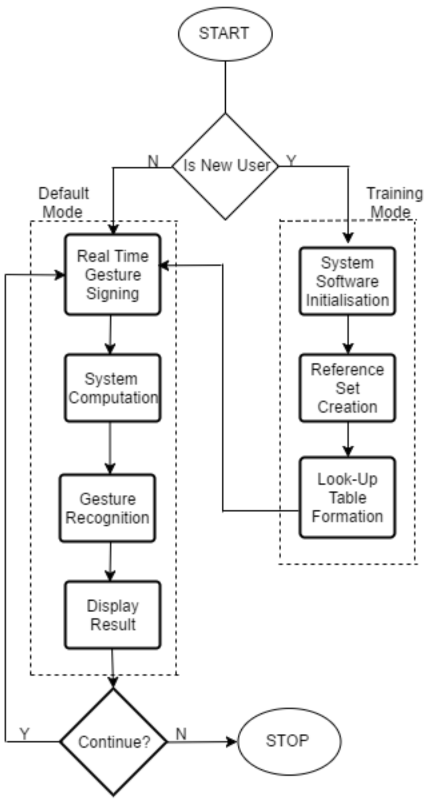

Fig. 3. Flowchart of the entire System Operation

1) Efficiency of the Recognition Software: The efficiencies of the system at various configurations of n_a and n_T were procured by testing the above explained *Data-Sets*. The average efficiency across the 3 users are presented in Table IV.

An instance of gesture recognition for a *Data-Set* (with $n_a = 5$ and a corresponding recommended $n_T = 2$) belonging to user 1, is represented in the form of a confusion matrix in Table V.

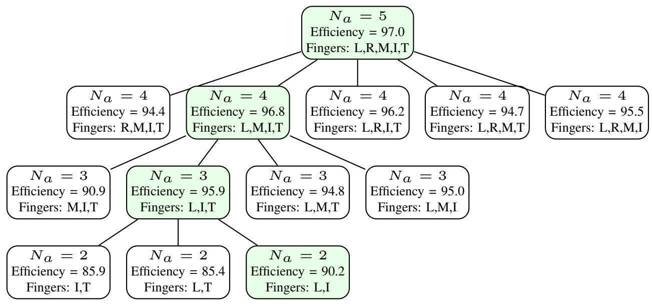

Fig. 4. Branch and Bound Tree for Minimization of Accelerometers; *Legend* - Fingers: 'L' - Little, 'R' - Ring, 'M' - Middle, 'I' - Index, 'T' - Thumb; User - 1; n_T = 1

2) Minimization of Accelerometers: The accelerometers were minimized based on a Branch-and-Bound approach for maximum efficiency. At each successive level of the minimization tree, the value of n_a reduces by one. The combination of sensors which produces maximum efficiency at each level is chosen and branched further to the next level, i.e., the minimized sensor is not considered for the subsequent levels. The process is iterated until n_a becomes 2, beyond which the resulting efficiencies proved to be minimal. This approaches reduces the number of combinations which require testing & validation and the *Data-Sets* were taken based on these combinations. The branching for the *Data-Sets* belonging to User-1 is shown in figure 4, where the nodes highlighted in green in the above figure were the fingers that exhibited maximum movement. These practical results were in coherence with the theoretical reduction of sensors (*Gesture-Set* specific), which ensues reduction based on least movement.

3) General Trend of Efficiency: Based on the saturation of the efficiency of the system with respect to n_T as seen in Table V, the curve was generalized into an equation as shown below.

$$\eta = \beta(1 - e^{-\alpha})$$

Where η is the efficiency of the system, β is the maximum attainable efficiency and α is a constant. The ranges of α corresponding to every n_A is shown in Table VI.

TABLE VI
RANGE OF α FOR VARIOUS MULTITUDES OF ACCELEROMETERS.

n_A	Range of α
5	3.19-7.05
4	3.39-6.21
3	3.89-7.30
2	4.80-5.80

Since the efficiency of the system saturates as n_T increases, an optimal value for n_T corresponding to every n_a was needed to be found. These values were termed as the *Recommended Number of Trials* as shown in Table III.

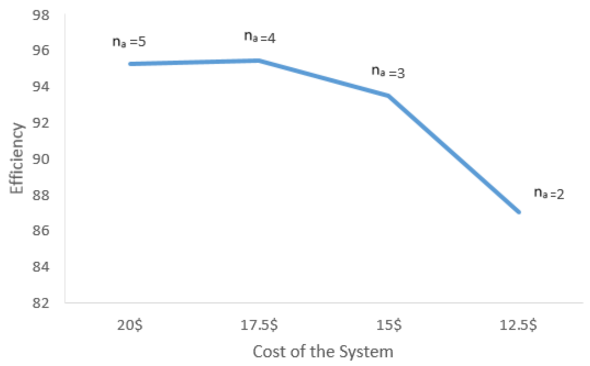

Fig. 5. Approximate cost of the system vs the efficiency of the system at recommended number of trials; for various values of n_a.

4) System Cost and Efficiency: The cost of the proposed system was realized at 12.5 USD for the $n_a = 2$ and at 20 USD for $n_a = 5$. Figure 5 shows the relationship between the cost of the system and the efficiency by plotting the efficiencies at various values of n_a and their corresponding recommended values of n_t. Also, this system lends to significant cost reduction from the prototype's cost upon commercial scaling.

V. CONCLUSION

Thus, a low-cost gesture recognition system using accelerometer sensors and an ATMega micro-controller was constructed and tested. The proposed system with minimal supervision requires a minimum of 24 seconds with a configuration corresponding to $\{n_a = 5, n_T = 2\}$ and a maximum of 84 seconds when $\{n_a = 2, n_T = 7\}$ for training, thus achieving an average run-time efficiency of 87.0% and 95.3% respectively. The cost of the system is realized at 20 USD when n_a=5 and and 12.5 USD when $n_a = 2$. It was concluded to use 3 accelerometers at its recommended number of 5 trials during training mode which takes 60 seconds, since this would be a justifiable trade-off between the cost, training and the efficiency of the system which results in an average run-time efficiency of 93.5%. The system is also compatable with wired or wireless communication, thus making it interoperable in other IoT platforms. The affordability and efficient working of the proposed system serves as a practical medium of communication for the audio-vocally impaired community and the low-income groups.

ACKNOWLEDGEMENT

The authors would like to acknowledge Solarillion Foundation for its support and funding of the research work carried out for the design and implementation of the gesture recognition glove prototype.

REFERENCES

[1] Fact Sheet No.300, Deafness and Hearing Loss, World Health Organization (WHO), March 2015, [Online]
Available: http://www.who.int/mediacentre/factsheets/fs300/en/

[2] S. Mitra and T. Acharya, "Gesture Recognition: A Survey," in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 37, no. 3, pp. 311-324, May 2007.

[3] D. J. Sturman and D. Zeltzer, A survey of glove-based input, IEEE Comput. Graph. Appl., vol. 14, no. 1, pp. 3039, Jan. 1994

[4] Rishikanth, C.; Sekar, H.; Rajagopal, G.; Rajesh, R.; Vijayaraghavan, V., "Low-cost intelligent gesture recognition engine for audio-vocally impaired individuals," Global Humanitarian Technology Conference (GHTC), 2014 IEEE , vol., no., pp.628,634, 10-13 Oct. 2014

[5] H. Sekar, R. Rajashekar, G. Srinivasan, P. Suresh and V. Vijayaraghavan, "Low-cost intelligent static gesture recognition system," 2016 Annual IEEE Systems Conference (SysCon), Orlando, FL, 2016, pp. 1-6.

[6] Rung-Huei Liang and Ming Ouhyoung, "A real-time continuous gesture recognition system for sign language," Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition, Nara, 1998, pp. 558-567.

[7] S. A. Mehdi and Y. N. Khan, "Sign language recognition using sensor gloves," Neural Information Processing, 2002. ICONIP '02. Proceedings of the 9th International Conference on, 2002, pp. 2204-2206 vol.5.

[8] J. M. Allen, P. K. Asselin and R. Foulds, "American Sign Language finger spelling recognition system," 2003 IEEE 29th Annual Proceedings of Bioengineering Conference, 2003, pp. 285-286.

[9] T. Chouhan, A. Panse, A. K. Voona and S. M. Sameer, "Smart glove with gesture recognition ability for the hearing and speech impaired," 2014 IEEE Global Humanitarian Technology Conference - South Asia Satellite (GHTC-SAS), Trivandrum, 2014, pp. 105-110.

[10] O. Sidek and M. Abdul Hadi, "Wireless gesture recognition system using MEMS accelerometer," 2014 International Symposium on Technology Management and Emerging Technologies, Bandung, 2014, pp. 444-447.